

adding the word “start” to the flow chart in FIG 10 so that all flow charts would be consistent. In addition, in FIG 10, note 8 in the right hand column, the symbol “!=” which means “not equal to” in programmer syntax has been changed to the more conventional mathematical symbol “ $\neq$ ” which means the same thing.

AMENDMENTS TO THE SPECIFICATION

At page 4, replace the section “DESCRIPTION OF THE DRAWINGS” as follows:

DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram showing the operational relationship of CCC 12 and FLU 14.

FIG. 2 is a detailed block diagram of the components of FLU 14.

FIG. 3 shows the detail for an entry or line in the cache in FLU 14.

FIG. 4 is a flow diagram depicting CCC operation and interaction with FLU 14.

FIG. 5 is a flow diagram depicting FLU operation and interaction with CCC 12.

FIG. 6 is an overall representation of the logic functions that would be compiled in the reconfigurable array to execute this block.

FIG. 7a shows one method of function compilation.

FIG. 7b shows an alternative method of function compilation.

FIG. 8 is a flow diagram for the exception routine.

FIG. 9 shows an alternative embodiment of CCC 12.

FIG. 10 shows a second alternative embodiment of CCC 12.

At page 10, line 11, replace the following paragraphs. FIG 4a should read FIG. 4 in several places.

FIG. 4[a] is a flow diagram depicting CCC 12 operation and interaction with FLU 14. The following is a legend with a numbered line by line description of FIG 4[a]. The numbered lines of FIG. 4[a] correspond to the numbered lines of the legend.

CCC FLOW LEGEND

1. Wait to go.
2. If attention signal asserted , take exception and deassert attention signal.
3. Set T to 0
4. If current instruction is transfer of control, T=1.
5. If current instruction writes to memory, send address and length to FLU (overwrite message).
6. Execute instruction and update state (registers, PC).
7. If next instruction accesses memory then M=1. If not M=0.
8. If not (T=1 and M=0), go to 4
9. Send GO signal to FLU and idle. Go to 1

In FIG. 4[a] the CCC is idle until given a GO signal (either from the FLU or as a function of reset). If the ATTENTION signal is raised, an exception is taken (the PC is saved somewhere and then set to the address of the exception routine). Execution continues on the CCC until the first instruction that does not access memory is reached after a transfer of control instruction is executed. This marks the entrance of a block of instructions that